532,542

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

22 APR 2005

(43) International Publication Date
13 May 2004 (13.05.2004)

PCT

(10) International Publication Number
WO 2004/040408 A2

(51) International Patent Classification[7]: G06F

(21) International Application Number: PCT/US2003/033509

(22) International Filing Date: 23 October 2003 (23.10.2003)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/421,285 25 October 2002 (25.10.2002) US

(71) Applicant *(for all designated States except US)*: **GRAND VIRTUAL, INC.** [US/US]; 100 Cambrige Park Drive, Cambridge, MA 02140 (US).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **UTIN, Daniil** [RU/US]; 161 Winding River Road, Needham, MA 02492 (US).

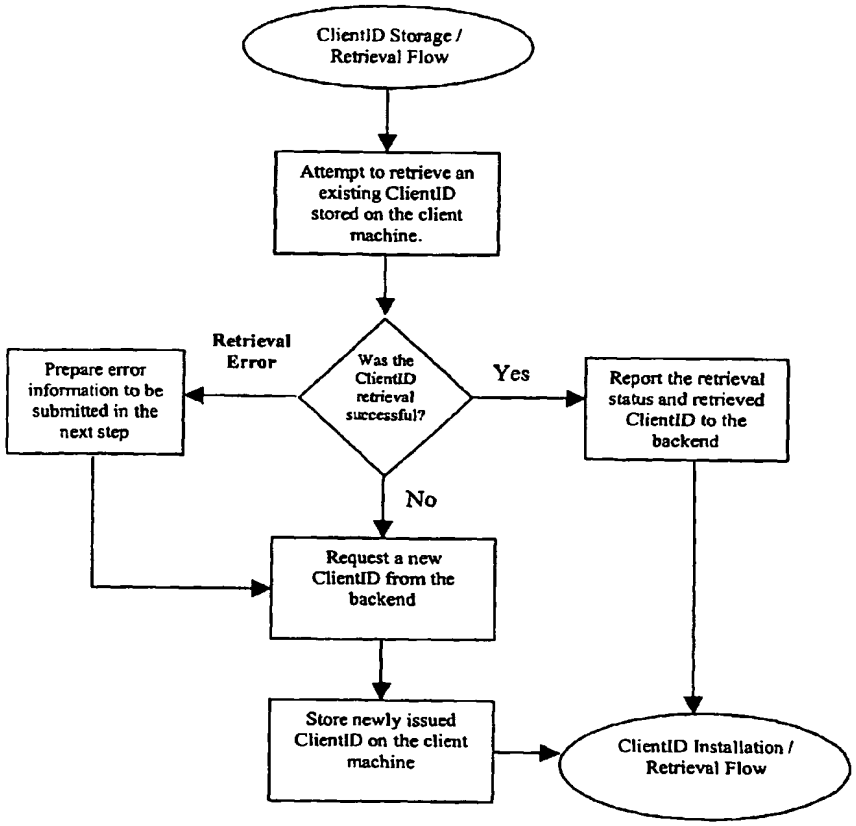
(74) Agent: **HENNESSEY, Gilbert, H.**; Fish & Richardson P.C., 225 Franklin Street, Boston, MA 02110-2804 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: FIXED CLIENT IDENTIFICATION SYSTEM FOR POSITIVE IDENTIFICATION OF CLIENT TO SERVER



(57) Abstract: A tamperproof ClientID system to uniquely identify a client machine is invoked upon connection of a client application to a backend. Upon initial connection, the backend issues a unique ClientID containing a checksum. The client application prepares at least two different scrambled versions of the ClientID and stores them in respective predetermined locations on the client machine. Upon subsequent connection to the backend, the client application retrieves and unscrambles the values at the two locations, verifies the checksums and compares the values. If the checksums are both correct and the values match, the ClientID value is sent to the backend, otherwise the client application sends an error code.

WO 2004/040408 A2

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

FIXED CLIENT IDENTIFICATION SYSTEM FOR POSITIVE IDENTIFICATION OF CLIENT TO SERVER

CLAIM OF PRIORITY

This application claims priority under 35 USC 119(e) to U.S. Patent Application Serial No. 60/421,285 filed on October 25, 2002, the entire contents of which are incorporated by reference.

TECHNICAL FIELD

This invention relates to security applications, and more particularly to identification of a user's computer.

BACKGROUND

Identification of a particular client computer system used for accessing a server is useful in secure applications where positive identification is desirable. In the past, systems for identifying client computers, browser cookies, for example, have had less than satisfactory capability of resisting tampering.

SUMMARY

A ClientID uniquely identifying a client machine is issued by the backend and stored on the client's machine upon first client application connection to the backend. On all subsequent connections, the client application retrieves the ClientID and sends it back to the backend. The ClientID mechanism includes features that make it very difficult for the user to remove or change the ClientID once it is set. In particular, according to the invention, this is accomplished by having the client application store at least two different scrambled versions of the ClientID in two separate locations in the client machine. Upon subsequent connection to the backend, the client application attempts to retrieve and unscramble the values at the two locations.

In the preferred embodiment, during the ClientID storage process, the backend generates a ClientID initially that contains a checksum and transmits it to the client application upon initial connection to the backend. The client application uses a first key to scramble the ClientID generating a first scrambled ClientID that is stored in the first predetermined location, for example the registry. A second key is used by the

client application to produce a second scrambled version of the ClientID that is stored in the second predetermined location, for example the system configuration file.

Upon subsequent connection of the backend, a retrieval process is invoked in which the client application retrieves the values at each location, unscrambles them using the respective keys, tests their checksums for verification, and compares the unscrambled values. If the checksums are both correct and the unscrambled values match, the retrieved ClientID is transmitted to the backend. Otherwise, the client application sends an appropriate error code to the backend.

The details of one or more embodiments of the invention are set forth in the accompanying drawings and the description below. Other features, objects, and advantages of the invention will be apparent from the description and drawings, and from the claims.

DESCRIPTION OF DRAWINGS

FIG. 1 is a flow and block diagram of the ClientID storage process.

FIG. 2 is a flow diagram of the ClientID retrieval process.

Like reference symbols in the various drawings indicate like elements.

DETAILED DESCRIPTION

ClientID is a special tag that uniquely identifies the client machine. Initially, the ClientID is generated by the backend and stored on client's machine upon first client application connection to the backend. On all subsequent connection, client application retrieves the ClientID and sends it back to the backend. Unlike browser cookies, the ClientID mechanism includes some special tamper-proof features that make it very difficult for the user to remove or change the ClientID once it is set.

Note: ClientID remains on the client's machine even after the client application is uninstalled. ClientID installation/retrieval occurs as a part of the client application startup process, as shown in FIG. 1, described in more detail below.

ClientID Storage Process

ClientID is stored in at least two undisclosed locations on the client machine (for example, in the registry and system configuration file). As shown in FIG. 1, the

ClientID value is encrypted on the backend and contains a checksum. The client application has an ability to verify whether the checksum is correct. This makes ClientID tampering much more difficult. In addition, prior to storing the ClientID in these two locations, the ClientID in each location is reversibly scrambled by the client application with two different keys. This makes it impossible to find the second ClientID location even if someone learns the first location and performs a search based on a value stored in the first location.

ClientID Retrieval Process

In the beginning of the ClientID retrieval process shown in FIG. 2, the client application attempts to retrieve and unscramble the values stored in both locations. Then it attempts to verify and compare these two values (if any were found).

All possible retrieval outcomes are listed below. Only the first two can be considered "normal", that is, should occur as a part of regular software usage. All other cases indicate that either someone is tampering with the ClientID mechanism or an Operating System malfunction/data corruption has occurred.

a. ClientID is not found in either of the two locations. This would normally happen when the software is started for the first time on the client machine. **Action**: request a new ClientID from the backend.

b. ClientID is found in both locations. The two values have a correct checksum and match each other. This should happen on the second and all subsequent client application launches. **Action**: report retrieved ClientID value to the backend.

c. ClientID is found in only one location. The value at that location has a correct checksum. **Action**: report retrieved ClientID to the backend along with error code #1 (see below for details)

d. ClientID is found in both locations. Only one value has a correct checksum. **Action**: report retrieved ClientID from the correct location to the backend along with error code #2 and a value from the other location.

e. ClientID is found in both locations. The two values have a correct checksum but do not match each other. **Action**: report retrieved ClientID value from

the first location to the backend along with error code #3 and a value from the second location.

f. ClientID is found in both locations. Values from both locations fail the checksum verification. **Action**: request a new ClientID from the backend, report error code #4 and values from both locations.

g. ClientID is found in only one location. The value at that location fails the checksum verification. **Action**: request a new ClientID from the backend, report error code #5 and a value from that location.

In cases c. through g. an error code along with some optional data is reported to the backend. That information is logged on the backend and, in conjunction with other data, like user IP, can be invaluable in detecting fraudulent activity. In cases c. through e. the error code and optional data are stored in the supplied ClientID record. In cases f. and g. that information is stored in the newly generated ClientID record.

A number of embodiments of the invention have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the invention. For example, more than two scrambled versions can be stored in respective locations. Accordingly, other embodiments are within the scope of the following claims.

WHAT IS CLAIMED IS:

1. A system for positively identifying a client machine running a client application to a backend, comprising

executing a ClientID storage process, including

upon connection by the client application to the backend, generating a unique ClientID containing a checksum at the backend for the client machine,

sending the ClientID to the client application,

reversibly scrambling the ClientID with the client application at the client machine and storing a first scrambled version of the ClientID at a first predetermined location on the client machine, and

reversibly scrambling the ClientID with the client application at the client machine and storing a second scrambled version different from the first version of the ClientID at a second predetermined location on the client machine.

2. The system of claim 1, further comprising executing a ClientID retrieval process with the client application when the client application subsequently attempts to connect to the backend, including

retrieving and unscrambling the values stored in both locations using the first and second keys,

running a checksum operation on the unscrambled values to verify that each has the correct checksum, and

comparing the two unscrambled values to see whether they match.

2. The system of claim 2, wherein the retrieval process executed by the client application further comprises

if the two unscrambled values retrieved from the two locations have the correct checksum and match each other, reporting the retrieved ClientID to the backend.

3. The system of claim 3, wherein the retrieval process executed by the client application further comprises

if the two unscrambled values retrieved from the two locations do not both have the correct checksum and match each other, reporting an error to the backend.

4. The system of claim 1, wherein the storage process further comprises encrypting the value of the newly generated ClientID at the backend and storing the encrypted version of the ClientID on the backend in a ClientID record.

5. The system of claim 1, wherein the storage process steps of scrambling use different first and second keys.

6. The system of claim 1, wherein one of the first and second locations is the registry.

7. The system of claim 1, wherein one of the first and second locations is the system configuration file.

8. The system of claim 1, wherein the first and second locations are the registry and system configuration file.

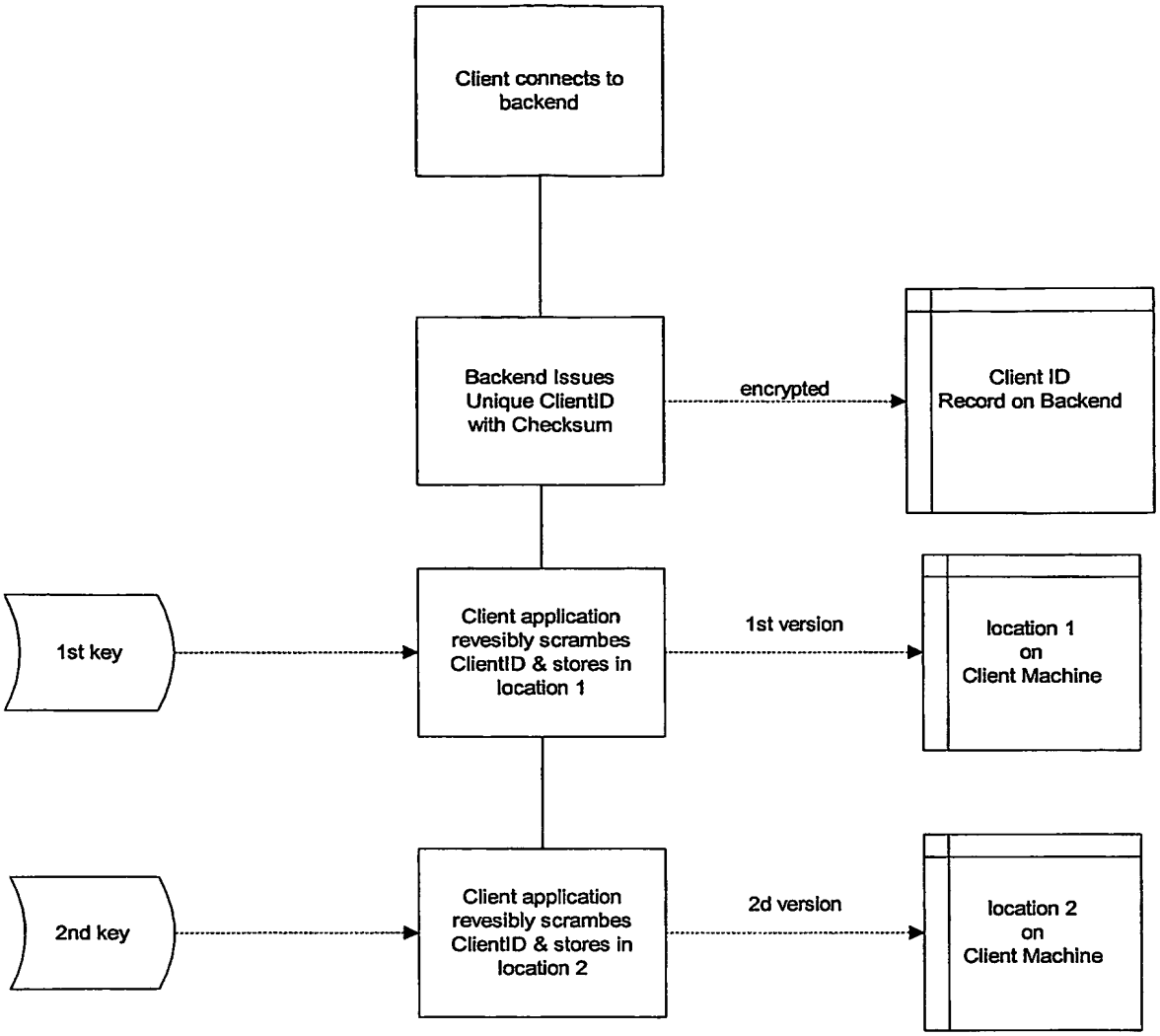

Client connects to backend
Backend Issues Unique ClientID with Checksum
encrypted
Client ID Record on Backend
1st key
Client application revesibly scrambes ClientID & stores in location 1
1st version
location 1 on Client Machine
2nd key
Client application revesibly scrambes ClientID & stores in location 2
2d version
location 2 on Client Machine

FIG. 1

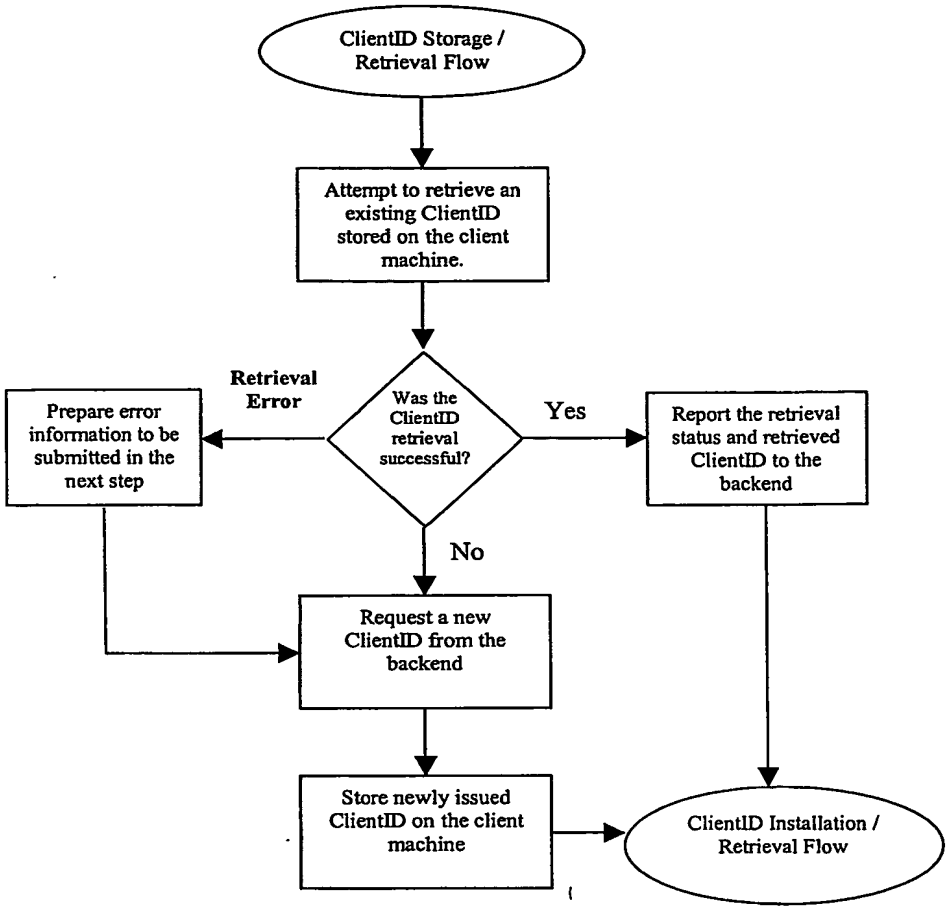

ClientID Storage / Retrieval Flow
Attempt to retrieve an existing ClientID stored on the client machine.
Was the ClientID retrieval successful?
Retrieval Error
Prepare error information to be submitted in the next step
Yes
Report the retrieval status and retrieved ClientID to the backend
No
Request a new ClientID from the backend
Store newly issued ClientID on the client machine
ClientID Installation / Retrieval Flow

FIG. 2